**Neo4j Data Load Script**

**#Defining constraints for the database**

CREATE CONSTRAINT ON (c:Case) ASSERT c.case\_no IS UNIQUE;

CREATE CONSTRAINT ON (co:Country) ASSERT co.name is UNIQUE;

CREATE CONSTRAINT ON (t:Type) ASSERT t.name IS UNIQUE;

CREATE CONSTRAINT ON (d:Decision) ASSERT d.status IS UNIQUE;

CREATE CONSTRAINT ON (j:Job) ASSERT j.title IS UNIQUE;

CREATE CONSTRAINT ON (s:SOC) ASSERT s.code IS UNIQUE;

CREATE CONSTRAINT ON (st:State) ASSERT st.name IS UNIQUE;

CREATE CONSTRAINT ON (ci:City) ASSERT ci.name IS UNIQUE;

CREATE CONSTRAINT ON (e:Employer) ASSERT e.name IS UNIQUE;

CREATE CONSTRAINT ON (w:Wage) ASSERT w.range IS UNIQUE;

CREATE CONSTRAINT ON (se:Sector) ASSERT se.name IS UNIQUE;

**#Loading Case and Country Nodes**

:auto USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM 'file:///US\_Permanent\_Visas.csv' AS line

WITH line, SPLIT(line.decision\_date, '/') AS date

MERGE (case:Case {case\_no: line.casenumber})

SET case.salary=TOINTEGER(line.pw\_amount\_9089)

SET case.month = TOINTEGER(date[0])

SET case.day = TOINTEGER(date[1])

SET case.year = TOINTEGER(date[2])

MERGE (country:Country {name: toUpper(line.country\_of\_citizenship) })

CREATE (case)-[:Citizenship]->(country)

**#Loading Decision and Status Nodes**

:auto USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM 'file:///US\_Permanent\_Visas.csv' AS line

WITH line

MATCH (case:Case {case\_no: line.casenumber})

MERGE (decision:Decision {status: toUpper(line.case\_status) })

MERGE (type:Type {name: toUpper(line.class\_of\_admission) })

CREATE (decision)-[:To]->(case)

CREATE (case)-[:Is]->(type)

**#Loading SOC Nodes**

:auto USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM 'file:///US\_Permanent\_Visas.csv' AS line

WITH line

MATCH (case:Case {case\_no: line.casenumber})

MERGE (soc:SOC {code: toUpper(line.pw\_soc\_code)})

ON CREATE SET soc.code=toUpper(line.pw\_soc\_code), soc.title=toUpper(line.pw\_soc\_title)

ON MATCH SET soc.title=toUpper(line.pw\_soc\_title)

CREATE (case)-[:Occupation]->(soc)

**#Loading Employer Nodes**

:auto USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM 'file:///US\_Permanent\_Visas.csv' AS line

WITH line

MATCH (case:Case {case\_no: line.casenumber})

MERGE (employer:Employer {name: toUpper(line.employer\_name)})

ON CREATE SET employer.name=toUpper(line.employer\_name), employer.address=toUpper(line.employer\_address\_1), employer.city=toUpper(line.employer\_city),

employer.state=(line.employer\_state), employer.postal\_code=toUpper(line.employer\_postal\_code)

ON MATCH SET employer.address=toUpper(line.employer\_address\_1), employer.city=toUpper(line.employer\_city),

employer.state=(line.employer\_state), employer.postal\_code=toUpper(line.employer\_postal\_code)

CREATE (case)-[:Sponsor]->(employer)

**#Loading Job Nodes**

:auto USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM 'file:///US\_Permanent\_Visas.csv' AS line

WITH line

MATCH (case:Case {case\_no: line.casenumber})

MERGE (job:Job {title: toUpper(line.job\_info\_job\_title) })

CREATE (case)-[:For]->(job)

**#Loading City and State Nodes**

:auto USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM 'file:///US\_Permanent\_Visas.csv' AS line

WITH line

MATCH (case:Case {case\_no: line.casenumber})

MERGE (city:City {name: toUpper(line.job\_info\_work\_city) })

MERGE (state:State {name: toUpper(line.job\_info\_work\_state) })

CREATE (case)-[:Location]->(city), (city)-[:Of]->(state)

CREATE (case)-[:Location]->(state)

**#Loading Wage Nodes**

:auto USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM 'file:///US\_Permanent\_Visas.csv' AS line

WITH line

MATCH (case:Case {case\_no: line.casenumber})

MERGE (wage:Wage {range: toUpper(line.Salary\_range) })

ON CREATE SET wage.range=toUpper(line.Salary\_range),

wage.level=toUpper(line.pw\_level\_9089)

ON MATCH SET wage.level=toUpper(line.pw\_level\_9089)

CREATE (case)-[:Has]->(wage)

**#Loading Sector Nodes**

:auto USING PERIODIC COMMIT

LOAD CSV WITH HEADERS FROM 'file:///US\_Permanent\_Visas.csv' AS line

WITH line

MATCH (job:Job {title: toUpper(line.job\_info\_job\_title) })

MERGE (sector:Sector {name: toUpper(line.us\_economic\_sector) })

CREATE (job)-[:In]->(sector)

**#Cypher queries for visualization**

**Query 1**

MATCH (case:Case)-[:Is]->(Type)

RETURN case.year, Type.name

**Query 2**

MATCH (case:Case)-[:Has]->(Wage)

RETURN case.year, Wage.range

**Query 3**

MATCH (case:Case)<-[:To]-(Decision)

RETURN case.year, Decision.status

**Query 4**

MATCH (case:Case)-[:Sponsor]->(Employer)

RETURN case.year, Employer.name

**Query 5**

MATCH (case:Case)-[:Sponsor]->(Employer)

RETURN case.year, Employer.city

**Query 6**

MATCH (case:Case)-[:For]->(Job)

RETURN case.year, Job.title

**Query 7**

MATCH (case:Case)-[:Citizenship]->(Country), (case:Case)<-[:To]-(Decision), (case:Case)-[:Sponsor]->(Employer), (case:Case)-[:Is]->(Type), (case:Case)-[:For]->(Job)

RETURN case.year, Country.name, Decision.status, Employer.name, Employer.state, Employer.city, Type.name, Job.title